苏州大学实验报告

院、系	计算机学院	年级专业	软件工程	姓名	朱金涛	学号	2327406014
课程名称	操作系统课程实践					成绩	
指导教师	王红玲	同组实验者	无		实验日期	2025 年 9 月 8 日	

实 验 名 称 Linux 系统操作及 C 语言编写

一、 实验目的

- 掌握 Linux Shell 常用命令的使用。
- 掌握 Linux 下 C 程序的编写、编译与运行方法。
- 掌握 gcc 编译器的编译过程，熟悉编译的各个阶段。
- 熟悉 Makefile 文件的编写格式和 make 编译工具的使用方法。

二、 实验内容

1. 练习常用的 Linux Shell 命令及命令选项，包括文件目录命令、备份压缩命令、重定向及管道命令等。要求熟练掌握下列命令的使用。

（1）改变及显示目录命令：cd、pwd、ls

（2）文件及目录的创建、复制、删除和移动命令：touch、cp、mv、rm、mkdir、rmdir

（3）显示文件内容命令：cat、more、less、head、tail

（4）文件查找命令：find、whereis、grep

（5）文件和目录权限改变命令：chmod

（6）备份和压缩命令：tar、gzip、bzip

2. 练习使用 gcc 编译器编译 C 程序并执行，编写 Makefile 文件，使用 make 工具编译程序并执行。

具体要求：

（1）编写简单的 C 程序，功能为在屏幕上输出“Hello gcc！”。利用该程序练习使用 gcc 编译器的 E、S、c、o、g 选项，观察不同阶段所生成的文件，即*.c、*.i、*.s、*.o 文件和可执行文件。

（2）编写一个由头文件 greeting.h、自定义函数文件 greeting.c、主函数文件 myapp.c 构成的 C 程序，并根据这三个文件的依赖关系编写 Makefile 文件。

三、 实验步骤和结果

1. 在 Linux 的命令行进行测试最基本的文件目录的操作指令:

（1）第一类 cd、pwd、ls 属于目录操作命令，pwd 准确输出当前目录绝对路径；cd 支持绝对 / 相对路径切换及 cd ..（回上一级）、cd ~（回家目录）；ls 通过-l（详细信息）、-a（隐藏文件）、-lh（人性化大小）按需展示内容，切换与查看功能均验证有效。

```
tt@tt-VMware-Virtual-Platform:~$ cd
公共/    视频/    文档/    音乐/    .cache/  .dotnet/ .pki/    .ssh/
模板/    图片/    下载/    桌面/    .config/ .local/  snap/    .vscode/
tt@tt-VMware-Virtual-Platform:~$ pwd
/home/tt
tt@tt-VMware-Virtual-Platform:~$ cd 下载/
tt@tt-VMware-Virtual-Platform:~/下载$ cd ..
tt@tt-VMware-Virtual-Platform:~$ cd 下载/
tt@tt-VMware-Virtual-Platform:~/下载$ cd ~
tt@tt-VMware-Virtual-Platform:~$ ls -l
总计 36
drwxr-xr-x 2 tt tt 4096 10月 13 14:56 公共
drwxr-xr-x 2 tt tt 4096 10月 13 14:56 模板
drwxr-xr-x 2 tt tt 4096 10月 13 14:56 视频
drwxr-xr-x 2 tt tt 4096 10月 13 14:56 图片
drwxr-xr-x 2 tt tt 4096 10月 17 11:38 文档
drwxr-xr-x 6 tt tt 4096 12月 14 23:12 下载
drwxr-xr-x 2 tt tt 4096 10月 13 14:56 音乐
drwxr-xr-x 2 tt tt 4096 10月 14 19:44 桌面
drwx------ 7 tt tt 4096 10月 16 11:41 snap
tt@tt-VMware-Virtual-Platform:~$ ls -a
```

（2）第二类 touch、cp、mv 属于文件/目录管理命令，touch 创建空文件，mkdir -p 创建多级目录；cp -r 递归复制目录，mv 实现移动与重命名；rm -r 删除目录，rmdir 仅删空目录。各操作均能正确完成文件 / 目录的创建、复制、删除及移动。

```
tt@tt-VMware-Virtual-Platform:~$ mkdir -p test
tt@tt-VMware-Virtual-Platform:~$ cd test/
tt@tt-VMware-Virtual-Platform:~/test$ touch a.txt
tt@tt-VMware-Virtual-Platform:~/test$ ls
a.txt
tt@tt-VMware-Virtual-Platform:~/test$ rm -r a.txt
tt@tt-VMware-Virtual-Platform:~/test$ ls
tt@tt-VMware-Virtual-Platform:~/test$ cd
tt@tt-VMware-Virtual-Platform:~$ touch a.txt
tt@tt-VMware-Virtual-Platform:~$ mv a.txt test/
tt@tt-VMware-Virtual-Platform:~$ cd test/
tt@tt-VMware-Virtual-Platform:~/test$ ls
a.txt
tt@tt-VMware-Virtual-Platform:~/test$ cd
tt@tt-VMware-Virtual-Platform:~$ rm -r test/
tt@tt-VMware-Virtual-Platform:~$ ls
公共  模板  视频  图片  文档  下载  音乐  桌面  snap
```

（3）第三类是内容显示命令，cat -n 带行号显示短文件；more 分页查看，less 支持滚动与搜索，交互性更强；head/tail 分别显示首尾内容，tail -f 实时监控新增内容，满足不同查看场景需求。

（4）第四类是查找命令，find 按路径、名称、类型、时间等多条件查找文件；whereis 快速定位命令的二进制及手册文件；grep -n/-i 实现带行号、忽略大小写的内容搜索，查找功能准确有效。

```
tt@tt-VMware-Virtual-Platform:~$ find test/
test/
test/hello.py
tt@tt-VMware-Virtual-Platform:~$ whereis ls
ls: /usr/bin/ls /usr/share/man/man1/ls.1.gz
```

（5）第五类是权限修改命令，数字法（如 chmod 755）按 r=4、w=2、x=1 设置权限，符号法（如 chmod u+x）为所有者 / 组 / 其他用户增减权限；-R 递归修改目录权限，权限调整均通过 ls -l 验证生效。

```
tt@tt-VMware-Virtual-Platform:~$ chmod u-x test_file.txt
tt@tt-VMware-Virtual-Platform:~$ ls -l test_file.txt
-rw-rwxr-x 1 tt tt 0 12月 22 16:18 test_file.txt
tt@tt-VMware-Virtual-Platform:~$ chmod a+r test_file.txt
tt@tt-VMware-Virtual-Platform:~$ chmod g+w test_file.txt
tt@tt-VMware-Virtual-Platform:~$ ls -l test_file.txt
-rw-rwxr-x 1 tt tt 0 12月 22 16:18 test_file.txt
```

（6）最后一类是备份压缩命令，gzip/bzip2 压缩文件，gunzip/bunzip2 解压；tar -cvf 打包，-zcvf/-jcvf 结合 gzip/bzip2 打包压缩，-xvf 解压，-tvf 查看压缩包内容，备份压缩流程完整可行。

```
tt@tt-VMware-Virtual-Platform:~$ echo "test1" > file1.txt
echo "test2" > file2.txt
echo "test3" > file3.txt
tt@tt-VMware-Virtual-Platform:~$ gzip file1.txt
tt@tt-VMware-Virtual-Platform:~$ ls -l file1.txt.gz
-rw-rw-r-- 1 tt tt 36 12月 22 16:36 file1.txt.gz
tt@tt-VMware-Virtual-Platform:~$ gunzip file1.txt.gz
tt@tt-VMware-Virtual-Platform:~$ cat file1.txt
test1
tt@tt-VMware-Virtual-Platform:~$ bzip2 file2.txt
tt@tt-VMware-Virtual-Platform:~$ ls -l file2.txt.bz2
-rw-rw-r-- 1 tt tt 46 12月 22 16:36 file2.txt.bz2
tt@tt-VMware-Virtual-Platform:~$ bunzip2 file2.txt.bz2\
> ^C
tt@tt-VMware-Virtual-Platform:~$ bunzip2 file2.txt.bz2
tt@tt-VMware-Virtual-Platform:~$ cat file2.txt
test2
```

2. （1）在命令行分别输入：gcc -E xxx -o xxx、gcc -S xxx -o xxx、gcc -c xxx -o xxx、gcc xxx -o xxx。

每次输入一行之后 ls 会发现目录中就会多出对应的.i、.s、.o、可执行文件。

其中-E 选项实现了仅执行预处理阶段，做一些比如将头文件导入、宏展开、删除注释等工作，不进行后续编译。-S 选项执行到编译阶段，生成汇编代码。-c 选项执行到汇编阶段，生成目标文件。-o 的作用是链接系统库文件或者指定输出文件的名称以及链接成可执行文件。最后-g 的作用是生成调试信息，用于调试工具。

结果图如下：

```
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ls
hello.c
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ gcc -E hello.c -o hello.i
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ls
hello.c  hello.i
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ gcc -S hello.i -o hello.s
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ls
hello.c  hello.i  hello.s
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ gcc -c hello.s -o hello.o
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ls
hello.c  hello.i  hello.o  hello.s
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ gcc hello.o -o hello
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ls
hello  hello.c  hello.i  hello.o  hello.s
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ gcc -g hello.c
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ls
a.out  hello  hello.c  hello.i  hello.o  hello.s
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ./hello
Hello, gcc!
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation$ ./a.out
Hello, gcc!
```

由图可见所有命令皆执行成功，且 hello 程序也能成功运行。

（2）编写好 myapp.c 和 greeting.c。因为有两个 c 文件，如果直接编译运行 myapp.c，那将导致程序无法链接到 greeting.c 文件，从而无法调用其中函数。解决方案便是同时编译两个

c 文件，在编译过程中会将两个目标文件链接成一个可执行文件 myapp。

结果图如下：

```
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation/ts002$ gcc myapp.c greeting.c -o myapp
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation/ts002$ ls
greeting.c  greeting.h  myapp  myapp.c
anders@anders-Legion-Y9000P-IRX8:~/OS_Operation/ts002$ ./myapp
Your Name, Please:Anders
Hello Anders!
```

由图可见程序成功运行。

编写的 Makefile 用于通过 gcc 编译器自动化构建 myapp 可执行程序，其中先定义了 CC（指定 gcc）、CFLAGS（设置-Wall 警告和- g 调试选项）、TARGET（目标可执行文件名）、SOURCES（源文件 myapp.c 与 greeting.c）、OBJECTS（由源文件转换的目标文件）、HEADERS（依赖头文件 greeting.h）等关键变量；随后通过目标规则定义了 myapp 依赖于所有目标文件、各目标文件依赖于对应源文件及头文件，并指定了编译（生成.o 文件）和链接（生成可执行程序）的具体命令；同时设置了 clean 伪目标（删除构建产物）和 rebuild 伪目标（先清理再重新构建），整体通过依赖管理实现仅编译修改文件，达成项目的高效自动化构建与维护。

代码如下：

```
#编译器和编译选项
CC = gcc
CFLAGS = -Wall -g
#目标可执行文件
TARGET = myapp
#源文件
SOURCES = myapp.c greeting.c
#目标文件(.o)
OBJECTS = $(SOURCES:.c=.o)
#头文件
HEADERS = greeting.h
#默认目标：生成可执行文件
$(TARGET):$(OBJECTS)
    $(CC) $(OBJECTS) -o $(TARGET)
#编译 myapp.o，依赖于 myapp.c 和 greeting.h
myapp.o: myapp.c $(HEADERS)
    $(CC) $(CFLAGS) -c myapp.c -o myapp.o
```

```
#编译 greeting.o, 依赖于 greeting.c 和 greeting.h
greeting.o: greeting.c $(HEADERS)
    $(CC) $(CFLAGS) -c greeting.c -o greeting.o
#清理生成的文件
clean:
    rm -f $(TARGET) $(OBJECTS)
rebuild: clean $(TARGET)
```

四、 实验总结

本次 Linux 系统操作及 C 语言编写实验顺利达成预设目标，系统掌握了 Linux Shell 常用命令、gcc 编译器使用及 Makefile 编写方法。在 Shell 命令实践中，通过实际操作验证了目录操作（cd、pwd、ls）、文件管理（touch、cp、mv、rm、mkdir 等）、内容显示（cat、more、less、head、tail）、文件查找（find、whereis、grep）、权限修改（chmod）及备份压缩（tar、gzip、bzip2）等命令的功能，能根据需求灵活运用命令选项完成对应任务，操作有效性均通过实际执行结果验证。在 gcc 编译部分，通过 -E、S、c、o、g 等选项分步执行预处理、编译、汇编、链接阶段，清晰观察到 *.i、.s、.o 及可执行文件的生成过程，同时解决了多 C 文件（myapp.c、greeting.c）编译时的链接问题，确保程序成功输出交互结果；编写的 Makefile 则通过定义编译器、编译选项、目标文件及依赖关系，实现了项目自动化构建，支持 clean 清理产物与 rebuild 重新构建，提升了编译效率，全面掌握了 Linux 环境下 C 程序的编写、编译与自动化构建流程。